Fault Tolerant Computation on Ensemble Quantum Computers

P. Oscar Boykin
Electrical Engineering Department
UCLA
Los Angeles, CA 90095
boykin@ee.ucla.edu

Tal Mor
Computer Science Department
Technion
Haifa 32000, Israel
talmo@cs.technion.ac.il

Vwani P. Roychowdhury
Electrical Engineering Department
UCLA
Los Angeles, CA 90095
vwani@ee.ucla.edu

Farrokh Vatan
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
farrokh.vatan@jpl.nasa.gov

Abstract

In ensemble (or bulk) quantum computation, all computations are performed on an ensemble *of computers rather than on a single computer. Measurements of qubits in an individual computer cannot be performed; instead, only expectation values (over the complete ensemble of computers) can be measured. As a result of this limitation on the model of computation, many algorithms cannot be processed directly on such computers, and must be modified. We provide modification of the fault tolerant quantum computation protocols to enable processing on ensemble quantum computers.*

1. Introduction

Quantum computing is a new type of computing which uses the properties of quantum mechanics to construct fast algorithms to solve several important problems. For example, Shor's quantum algorithm [22] for factoring large numbers is exponentially faster than any known classical algorithm. Similarly, by utilizing Grover's algorithm [12], it is possible to search a database of size N in time $O(\sqrt{N})$, compared to $O(N)$ in the classical setting.

Nuclear Magnetic Resonance (NMR) computing, first suggested by Cory, Fahmy and Havel [10], and by Gershenfeld and Chuang [11], is currently one of the most promising implementations of quantum computing. Several quantum algorithms involving only few qubits have been demonstrated in the laboratory setting [10, 11, 9, 13, 17, 23]. In such NMR systems, each molecule is used as a computer. Different qubits in the computer are represented by spins of different nuclei. Many identical molecules (in fact, a macroscopic number) are used in parallel; hence, there is an ensemble of quantum computers. This model is called the ensemble or bulk quantum computation model. In such bulk models, each operation is applied to each computer in the ensemble. Qubits in a single computer cannot be measured, and only expectation values of each particular bit over all the computers can be read out.

The impossibility of performing measurements on the particular qubits of individual computers causes severe limitations on ensemble quantum computation. In particular, for quantum cryptography tasks, ensemble quantum computers appear to be useless. It was generally assumed that rather simple strategies of delaying (or avoiding) measurements can be used to bypass these limitations, in order to enable the implementation of *all* quantum algorithms. We show here that the existing strategies are *insufficient* for fault tolerant computation, and we develop novel strategies that resolve this problem.

In this paper, we restrict ourselves to issues related solely to the **ensemble–measurement problem**. The results here are vital for bulk computation; in addition, the specific results obtained regarding universal and fault tolerant sets of gates are also important for other implementations of quantum computing devices where reducing the number of measurements required for computation is desired.

Proceedings of the 2004 International Conference on Dependable Systems and Networks (DSN'04)
0-7695-2052-9/04 $ 20.00 © 2004 IEEE

2. The measurement in ensemble quantum computation

The measurement process in quantum mechanics can be described simply as follows: To measure the state of a qubit, say $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ in the computation basis $(|0\rangle; |1\rangle)$, one measures the Hermitian operator (the observable)

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

to get the outcome $\lambda_0 = 1$ with probability $|\alpha|^2$ and $\lambda_1 = -1$ with probability $|\beta|^2$. In an NMR ensemble model, the corresponding qubit in every computer is measured simultaneously, resulting in the expectation value, i.e., the outcome of the measurement is a signal of strength proportional to $|\alpha|^2 - |\beta|^2$.

The inability to measure bits in individual computers precludes using measurements to reset bits. A simple way to reset a bit is to measure it and flip it if the outcome is $|1\rangle$. Since each computer in the ensemble will have a different outcome, this is impossible on an ensemble computer. Algorithmic cooling has been proposed to reset bits in ensemble computing settings[20, 7].

The measurement process lies at the heart of all quantum information processing and computing protocols and algorithms, and hence, needs to be carefully addressed in any proposed implementation scheme. Clearly, when the outcome of a measurement is expected to be the same on each of the computers, the ensemble measurement is as good as the standard (single computer) measurement. Usually, this is not the case. In fact, to the best of our knowledge, the following two protocols cannot be implemented on an ensemble quantum computer due to the measurement issue:

Random number generator (RNG): One can easily create an RNG Using a single qubit. To create a binomial probability distribution with parameter p, one prepares a state $\sqrt{p}|0\rangle + \sqrt{1-p}|1\rangle$, and measures in the computational basis to obtain the desired RNG. This, as far as we know, cannot be done on an ensemble quantum computer, where only the expectation value $p\lambda_0 + (1-p)\lambda_1$ can be monitored.

Teleportation: Standard teleportation can easily be performed on a three qubit quantum computer. Strictly speaking however, it cannot be performed on an ensemble quantum computer. This is because a direct Bell-state measurement of the ensemble quantum computer is computationally useless: each computer will yield a random result (of the Bell measurement), and on average the outcome is $(1/2)\lambda_0 + (1/2)\lambda_1$ for each of the two measured qubits; hence, there is no way to decide how to rotate the third qubit in each individual computer. Yet, a "fully-quantum teleportation" of the type suggested in [8] can be, and has been [17], performed on an ensemble quantum computer: in this fully-quantum teleportation, the measurement of an individual computer is never monitored, and a classically-controlled rotation of the third qubit is replaced by a quantum control operation, in which the control qubits dephase before being used.

To better appreciate the measurement problem it is instructive to review the basic anatomy of a quantum computer. At a very high level, a quantum algorithm can be described as a set of unitary transformations to be applied to an n-qubit system, followed by a measurement of m of the qubits to obtain a classical m-bit output. The m-bit classical output is either (i) One of many possible "correct" or " good" answers. For example, a database search using Grover's algorithm will return one of the entries satisfying the query. If there are multiple possible query hits, then every time the algorithm is run, it will return any one of the hits with equal probability. Thus, in ensemble quantum computing, even though all the computers do identical operations, they will have different outcomes after the measurement process, and one cannot get one correct answer by the reading process we described. (ii) The m-bit output is either the "desired" or "good" solution or it is a spurious or "bad" candidate, and the whole process has to be repeated again. For example, in Shor's factorization algorithm the measurement process yields an integer, which a set of classical operations can process to verify whether it yields the correct answer (i.e., the order of the input integer) or not. The probability that the output will yield the correct answer is such that one is guaranteed to obtain such an output in a few number of repeated executions of the algorithm. Again, in the case of ensemble quantum computing different computers will yield different results, and the task of identifying the correct answer needs to be addressed.

If the role of measurements was restricted to only the two cases described above then it is not that difficult to come up with solutions to the measurement problem for ensemble quantum computation. However, there are other "hidden" uses of the measurement process, particularly involving error correction and fault tolerance, during the execution of the algorithm (i.e., during the part that we broadly described as involving only unitary operations) that are a lot harder to address. In this paper, we solve the problem of performing fault-tolerant ensemble quantum computation that had remained unanswered.

We next briefly review some of the strategies that

one can adopt to successfully overcome the measurement problem in ensemble quantum computers, which in turn will allow us to point out why the existing approaches will not work for the fault tolerant computing problem, addressed in this paper.

1. Let us consider the case, where the measurement output is processed/used in one of the following ways (i) The classical output is processed to determine whether it yields the "desired" outcome or not. For example, in Shor's algorithm the output is processed to determine if it yields the order or not. (ii) The classical output is used to perform specific unitary operations on the rest of the qubits, as in the case of quantum error-correction.

   This case was recognized in the seminal work of Gershenfeld and Chuang [11]. The most effective scheme is to simply delay (or even avoid) the measurements, and incorporate the post-measurement processing step into the quantum algorithm, as a controlled operation.

   In the case of Shor's algorithm one can perform the classical computation necessary to verify whether the candidate output generates a correct order or not. After these operations, the answer is yes or no, and all the computers with the "yes" answer, will have the *same answer*, i.e., the desired order. However, the computers with "no" answers, will have different answers after the post-measurement processing steps. We can solve the problem posed by the interference due to the "bad" candidates by replacing bad results with random data, which will not interfere on average with the reading of the "good" result. Previous work by Gershenfeld and Chuang [11] noted that Shor's factorization algorithm can be implemented on ensemble quantum computers, by incorporating the post-measurement processing steps into the quantum algorithm. However, Shor's algorithm on ensemble computers suffers also from problem of generating a number of "bad" solutions, and hence the modified algorithm suggested in [11] is not sufficient. The algorithm requires a further modification (the randomizing-bad-results strategy) in order to work in the general case. Alternatively, one might be able to control-repeat the computation in case the classical verification part shows that the algorithm yielded a bad output. Unfortunately, such strategy is not easily implementable and cannot be easily justified; furthermore it leads to a much longer computation process, and hence to higher sensitivity to errors.

2. The algorithm has more than one correct final outcome and the measurement process directly yields one of the correct solution [e.g., Grover's search algorithm with several solutions that we already pointed out]. First, note that for a measurement model where all the 2^n states of an n-qubit system can be distinguished, the multiple-solutions case is not a problem. However, such a scheme is not practical for any algorithm involving even tens of qubits, and the exponential resolution requirement makes it no better than a classical computer. [6] suggests modifications for scalable measurement processes (i.e., where measurement is done one qubit at a time) so that problem due to the multiple search outputs can be resolved. One solutions involves making multiple searches on the same computer and then sorting the results. This way with high probability the computers will have the same sorted list.

It is fair to say that it was generally assumed that the strategy of delaying measurements (as in case (1) above) could be used to save *all* quantum algorithms. However, as pointed out in both the above-mentioned cases, delaying measurements by control operations solves only some of the problems, and both the Shor and Grover algorithms require strategies that are very different and were not considered explicitly in prior literature. The measurement problem becomes even more acute in the case of fault-tolerant computations, which is the subject of this paper. The schemes proposed so far for quantum fault tolerant computation provide an incomplete set of gates, i.e., a set of gates that is not universal for quantum computation. In order to complete the set to a universal set, the schemes use interactions with ancilla qubits, which are then measured [21, 16, 19]. Each such measurement is followed by an application of a unitary operation, U_j, that depends on the outcome of the measurement (j). A direct scheme for removing such measurements (followed by the required unitary operations U_j), and replacing them by controlled operations, $\Lambda(U_j)$, will not in general be realizable. This is because, $\Lambda(U_j)$ might not be realizable by the incomplete set of fault tolerant gates. For example, if one attempts to remove measurements in Shor's scheme for fault tolerant realization of Toffoli gate [21], then the corresponding controlled operations would itself require Toffoli gates! We believe that this issue was not explicitly addressed in previous works. The rest of this work will describe how an analysis of error propagation and a careful usage of classical reversible circuits can allow one to delay measurements in a fault tolerant manner, and allow for fault tolerant NMR quantum computing.

3. Review of Fault Tolerant Quantum Computing

The idea of quantum fault tolerant computation [21, 3, 16, 14, 19] can be described briefly as follows. Suppose that we have a (noise-less) quantum circuit C which we want to simulate by a noisy quantum computer. On the noisy quantum computer, instead of circuit C we perform a fault tolerant circuit $\widetilde{C}$. The physical bits $|0\rangle$ and $|1\rangle$ are replaced by logical bits $|0\rangle_L$ and $|1\rangle_L$, where these are some entangled states of a block of physical qubits. While C operates on physical qubits representing the data, in the circuit $\widetilde{C}$ all operations are performed on logical qubits which are error-correction-encoded data, i.e., each data qubit or a set of data qubits is represented as a block of qubits that belongs to some quantum error–correcting code. Then each operation of C performed by a gate g_j is simulated by a procedure (sub-circuit) $\widetilde{g_j}$ in the circuit $\widetilde{C}$ such that in $\widetilde{g_j}$ each computation transforms codewords to codewords. In order to avoid accumulation of errors, after each computation in $\widetilde{g_j}$ a correction procedure is performed to correct any error that is introduced in that computation. Thus, in the fault tolerant circuit $\widetilde{C}$ each computation step is followed by a correction step.

The operations on the encoded qubits introduce a large number of additional gates and qubits, and, unless one is careful, it is possible that more errors are introduced than can be corrected by the code. To avoid any such catastrophic accumulation of errors, it is desirable that the operations in the fault tolerant circuits prevent "spreading of errors" by making sure that each gate error causes at most a single error in each block. It is useful now to review how errors propagate in quantum circuits. For example, consider the CNOT (controlled–not) gate which performs the operation $|a\rangle_c |b\rangle_t \mapsto |a\rangle_c |a \oplus b\rangle_t$ in the computation basis; for the rest of this paper, we shall drop the subscripts c (control) and t (target) and designate the control bit as the one on the left side. Clearly, applying the CNOT operation from one bit to many target bits can propagate one bit error from the control bit to all the target bits. On the other hand, applying CNOT from many control bits to one target bit can propagate one phase error from the target bit to all the control bits. It is easy to observe this "back" propagation of the phase errors: if a phase error happens on the second (target) qubit in the state $(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle)$ and a CNOT is applied after, we will get

$$\begin{aligned} &(|0\rangle + |1\rangle) \otimes (|0\rangle - |1\rangle) \\ &\xrightarrow{\text{CNOT}} \quad |0\rangle \otimes (|0\rangle - |1\rangle) + |1\rangle \otimes (|1\rangle - |0\rangle) \\ &= \quad (|0\rangle - |1\rangle) \otimes (|0\rangle - |1\rangle) \end{aligned}$$

which results in a phase error in the control qubit. Hence, fault tolerant computation requires that this gate be applied only in the case where the control qubit $|a\rangle$ and the target qubit $|b\rangle$ belong to different blocks.

This error-propagation phenomenon is also true for other controlled operations, and this motivated a *sufficient* condition for fault tolerance: only perform bit-wise or transversal[1] operations on qubits within a code. It is, however, *not* a *necessary* condition for fault tolerance, and careful constructions may allow one to apply control gates from many control bits onto one target bit, without destroying the fault tolerant computation.

Therefore, to achieve a quantum fault tolerant computation, it is enough to show that a *universal* set of quantum gates can be constructed with only bit-wise and transversal operations on a quantum code. Quantum fault tolerant schemes usually (see, e.g., [21, 19]) depend on measurements to ensure that the set of the operations permissible on encoded data is actually a universal set. Recall that we cannot depend on measurements in ensemble computers, but must still create a universal set to achieve fault tolerance. Some of the gates in the universal set do not require measurements, e.g., the operations H, $\sigma_z^{1/2}$, and CNOT. [For a class of codes called CSS codes [21], H, σ_z, and CNOT can simply be achieved by performing the same gate bit-wise on the individual qubits (e.g., H is achieved on code words via applying H on individual qubits), while the bit-wise $\sigma_z^{1/2}$ yields a $\sigma_z^{-1/2}$ logical gate, hence requires an additional step of bit-wise σ_z, to yield the desired logical gate.] In previous works [21, 4, 5] at least one gate in the universal set requires measurements. That's bad for ensemble computers. We now present tools that will allow us to create a measurement-free universal quantum fault tolerant set of gates.

4. Measurement-free Quantum Fault Tolerant Gates

There is always a simple scheme that potentially allows one to postpone measurements of ancilla qubits in quantum computation. Unfortunately, the simple scheme never works in the case of generating universal fault tolerant gates. In fault tolerant computation and error recovery, often a measurement is followed by

1 By transversal operations, we mean operations that act on at most one qubit in any code block. For instance, a gate applied from the first bit of one codeword and the first bit of a second codeword, and second bit of one codeword and a second bit of a second codeword, and so fourth.

an operation U_j [2] *if* the outcome of the measurement is 1. As explained in Section 2, the scheme for delaying the measurement can be successfully implemented only if the measurement followed by a U_j operation can be replaced by controlled-U_j (denoted $\Lambda(U_j)$) in the set of *available* measurement–free operations; i.e., control operations which can be implemented on encoded data fault tolerantly and directly without using any measurements. However, in the schemes proposed so far, the required control operations $\Lambda(U_j)$ are not implementable in a direct fault tolerant manner. For instance, in Shor's fault tolerant set of gates [21], a measurement is required for the preparation of a Toffoli gate, but a Toffoli gate is required if we want to delay that measurement. This is because the measurement is followed by a CNOT operation, and hence can only be replaced by a controlled–CNOT, which is a Toffoli gate. This seems like a catch-22 situation! [3]

The solution comes from the vital observation that some operations need protection only from the bit errors, and do not need to use full quantum codes: by replacing the quantum ancilla (in a logical basis $|0\rangle_L$ and $|1\rangle_L$) by a "classical ancilla" in a "classical" basis $|\vec{0}\rangle = |0\cdots0\rangle$ and $|\vec{1}\rangle = |1\cdots1\rangle$, we can use the classical ancilla to perform $\Lambda(U_j)$ in a fault tolerant manner. This can be done in the two cases where the the Toffoli gate is required for Shor's fault tolerant set of gates, and the $\Lambda(\sigma_z^{1/2})$ gate required for the basis of [4]. One can interpret the classical basis as the classical repetition code. We call the ancilla in these states "classical" since we are not concerned with phase errors on these bits. A classical error-correction code can correct bit errors in the classical ancilla. Despite the fact that phase errors are not corrected in the classical ancilla, we found that the use of such a classical ancilla is still good enough for our purpose.

4.1. Replacing Measurements of Encoded Ancilla Qubits

In the following, we shall replace the measurement of the quantum ancilla followed by the operation U acting on the quantum data, by a sequence of operations: we copy the two basis states of a quantum ancilla into a classical ancilla, we perform classical error correction on the classical ancilla, and we use the classical ancilla as a control bit for performing the operation $\Lambda(U_j)$ with the quantum data as the target bit.

The measurement of the quantum ancilla in the original protocol can be done as follows [19]: measure each of the physical qubits, and perform a classical error correction on the outcome of this measurement to determine the state of the ancilla. For example, if the 7-bit CSS code [21] is used to encode data, then a measurement will yield a (possibly corrupted) codeword of a classical 7-bit Hamming code. After classical error correction, if the parity of the codeword is even, then the ancilla has collapsed to the state $|0\rangle_L$; otherwise, it has collapsed to the state $|1\rangle_L$. Classical error correction is enough to protect the output bit b, because phase errors before a measurement will not change the outcome probabilities.

In Figure 1, we represent a circuit that computes operation $\mathcal{N}_1$ for the seven–bit CSS code, where $\mathcal{N}_1$ stands for Eq.(1) with only one bit of the classical ancilla. The ancilla bits labeled syndrome are used to prevent the spread of one bit error from the quantum ancilla into the classical bit. These bits are exactly the parity check of the syndrome of the 7-bit Hamming code. Only two errors (in any of the inputs, the gates or the time steps) shall yield an error in the classical bit.

The circuit $\mathcal{N}_1$ flips the bit b if the quantum ancilla (acting here as a control bit) is $|1\rangle_L$, and does nothing otherwise. This circuit operates properly as long as there is up to one bit error in the quantum data (there can actually be an unlimited number of phase errors). Note that phase errors in the lower part will spread to the quantum ancilla. This is of no consequence, however, since the quantum ancilla never interacts with the quantum data in later stages. Bit errors in the quantum ancilla are important, since the process is repeated n times; hence, bit errors, created in the quantum ancilla at initial stage of $\mathcal{N}_1$, will spread errors into the next bits of the classical ancilla. Fortunately, bit errors are not transmitted from the classical to quantum section, and the quantum ancilla cannot be disturbed by a bit error in bits of the classical ancilla or the syndrome ancilla.

As a step toward removing the measurement from the original protocol, we propose a new gate that copies an encoded quantum ancilla word onto a classical ancilla:

$$\mathcal{N}: \begin{cases} |0\rangle_L \otimes |\vec{0}\rangle & \longrightarrow \quad |0\rangle_L \otimes |\vec{0}\rangle, \\ |0\rangle_L \otimes |\vec{1}\rangle & \longrightarrow \quad |0\rangle_L \otimes |\vec{1}\rangle, \\ |1\rangle_L \otimes |\vec{0}\rangle & \longrightarrow \quad |1\rangle_L \otimes |\vec{1}\rangle, \\ |1\rangle_L \otimes |\vec{1}\rangle & \longrightarrow \quad |1\rangle_L \otimes |\vec{0}\rangle. \end{cases} \tag{1}$$

2 U_j can be performed fault tolerantly using the given, non-universal, set of operations.

3 Similarly, in the fault tolerant universal set of gates suggested in [4], the generation of the $\sigma_z^{1/4}$ gate without measurements leads to a catch-22 problem; a $\sigma_z^{1/2}$ gate (which follows the measurement) needs to be replaced by a $\Lambda(\sigma_z^{1/2})$ gate, which is not available as long as the $\sigma_z^{1/4}$ gate is not available.

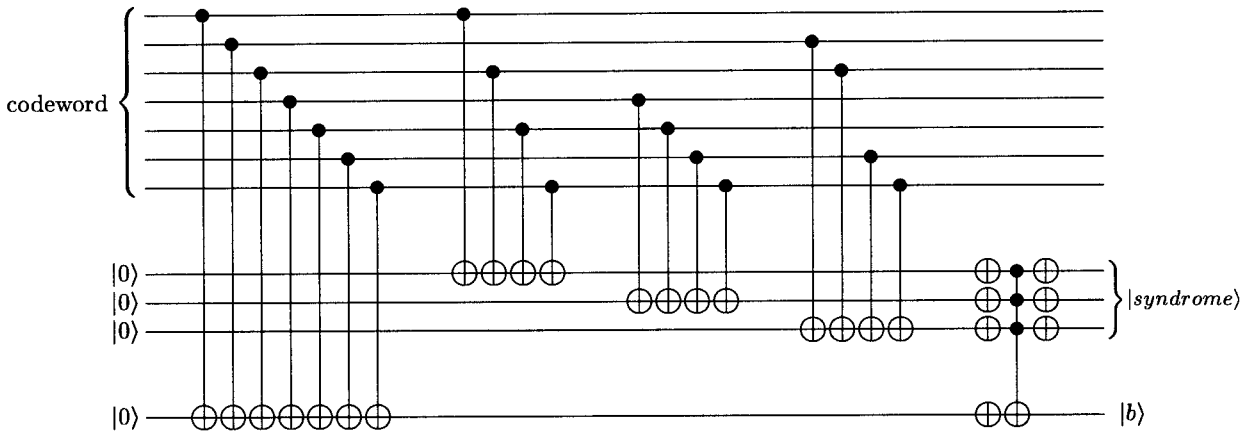

Figure 1. The operation $\mathcal{N}_1$. Note that the circuit shows the generation of only one classical target bit $|b\rangle$; the operations on the last bit have to be repeated to generate multiple target bits.

Let $\mathcal{N}$ be a unitary operation that implements the above transformation. In the next section we show that the complete $\mathcal{N}$ operation can be done fault tolerantly. In sections 4.4 and 4.5, the operation $\mathcal{N}$ will enable us to construct gates for universal fault tolerant computation, without measurement.

4.2. The operation $\mathcal{N}$: quantum-to-classical controlled–NOT

The $\mathcal{N}$ operation "copies" the encoded quantum bit onto the encoded classical ancilla. The repetition code can only correct for bit errors in the classical ancilla, but one must be sure that the classical ancilla can still be used to perform $\Lambda(U_j)$ without putting the quantum data in jeopardy. Perhaps counter-intuitively, this is not a problem, since phase errors are transmitted from *target* bit to *control* bit, hence cannot be transmitted from the classical ancilla (control) to the quantum data (target). This leads to the most interesting aspect of our scheme: the data in the classical repetition code, or any classical function of this data, *can* act as control bits in a bit-wise controlled-U operation onto quantum data.

In the complete $\mathcal{N}$ circuit, the $\mathcal{N}_1$ computation on the bottom four bits is repeated at most n times, where n is the number of qubits in a codeword. At each repetition stage, the syndrome bits are discarded, and another bit b_i is created ($i \in 1 \ldots n$). In principle, the syndrome bits could be ignored, reset, or measured. These bits will not affect the operation beyond their use as a form of error detection in the codeword. The bits b_i are then corrected (to yield the classical 0 or 1) using a majority vote.

In order to reduce the number of operations (and hence improve the fault tolerant threshold), we only need to use a repetition code that will successfully recover from k' errors. Once this number k' is equal to, or greater than, the number of errors, k, that the *quantum* code can correct for, we may stop. For a probability p of an error (per gate, per input bit, and per delay line), the resulting error rate of this circuit is $\mathcal{O}(p^2)$, as required for fault tolerant computation. The threshold can easily be calculated by counting the potential places for two errors, and the threshold can be much improved by enhancing the parallelism, and by repeating $\mathcal{N}_1$ only $2k+1$ times (e.g., with the 7-bit quantum code, that is $n = 7$, which corrects $k = 1$ error, it is enough to repeat the circuit 3 times, correct the outcome using a majority vote, and then copy the result into seven bits).

Later, in sections 4.4 and 4.5, we show cases where, indeed, the operations between the classical ancilla and the quantum data can be performed bit-wise, while the same operations cannot be performed bit-wise between quantum ancilla and the quantum data (as the naive solution of delaying measurements would have suggested).

Note that the quantum data may add phase errors to the repetition code, but that is of no concern to us, since the classical repetition code also loses phase coherence in the measured case. If there are t bit errors in the repetition code, it will result in t errors in the quantum data. Fortunately, bit errors *are* corrected in the repetition code. Hence, *the operation $\mathcal{N}$ enables one to create universal bases without measurement.*

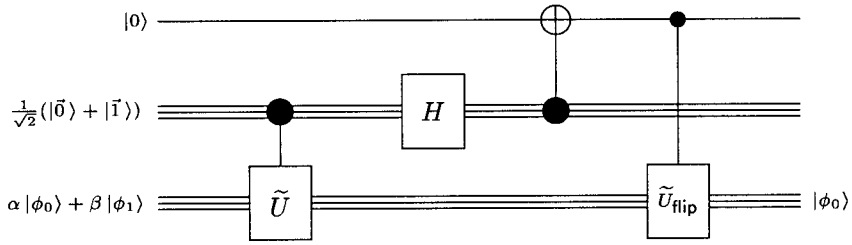

Figure 2. Preparing an eigenvector.

4.3. Creating the special states required for fault tolerant universal computation, without using a measurement

In sections 4.4 and 4.5 we describe how to construct gates to produce a universal set without measurement. In both of those sections, we will make use of "special states" which enable the construction of the gate. In this section we describe a general method to produce these special states under general circumstances. Once presented, the descriptions in sections 4.4 and 4.5 become much simpler.

Assume that a quantum code of length n is used for encoding data. Suppose that $U \in \mathrm{U}(2^l)$ [for our purpose it is enough to consider up to three qubits ($l = 3$) operations], and $\widetilde{U} = U^{\otimes n}$ is the unitary operation on the codewords obtained by applying U bit-wise. Suppose that $\widetilde{U}$ has eigenvectors $|\phi_0\rangle$ and $|\phi_1\rangle$ such that

$$\widetilde{U}|\phi_0\rangle = |\phi_0\rangle \qquad \text{and} \qquad \widetilde{U}|\phi_1\rangle = -|\phi_1\rangle .$$

Then the quantum circuit in Figure 2 outputs the eigenvector $|\phi_0\rangle$ if the input state is $\alpha|\phi_0\rangle + \beta|\phi_1\rangle$ for any α, β. In figure 2, $\widetilde{U}_{\text{flip}}$ is a unitary operation that maps $|\phi_0\rangle$ on $|\phi_1\rangle$ and vice versa. The operations $\Lambda(\widetilde{U})$ (i.e., the controlled–$\widetilde{U}$), and H are applied bit-wise.

This scheme is practical if it is possible to prepare a state $\alpha|\phi_0\rangle + \beta|\phi_1\rangle$, where the values of α and β do not matter. In this circuit the first line is a single parity bit, and each of the second and third inputs are blocks of n qubits, containing the cat-states lines and the special state lines, respectively. The third gate, the CNOT gate which we call here P, is a parity gate which calculates the parity of the cat-state lines and puts the result in the parity bit. This is done by a sequence of CNOTs from each control bit onto one target bit. The figure only demonstrates the creation of one parity bit $|\phi_0\rangle$ in an unprotected manner as far as a bit error in the parity bit is concerned. The real circuit is a bit different: The operations $\Lambda(\widetilde{U})$, H and P, are repeated n times, each time with fresh cat-states and a fresh parity bit (but on the same special state's lines). Then a majority vote is calculated on the parity bits, in order to reduce the probability that an error in a cat state or in the parity bit will ruin the result. Then the n parity results are corrected, so that the probability of two errors becomes low [that is, of order $\mathcal{O}(p^2)$]. Finally, the parity result is used to control $\widetilde{U}_{\text{flip}}$ in a bit-wise manner, so that the special state is created via a fault tolerant operation.

4.4. Fault tolerant ${\sigma_z}^{1/4}$ without measurement.

We show here a modified version of the original method for implementing ${\sigma_z}^{1/4}$ on codewords [4] which does not use measurements. Using the method described in Section 4.3, we need to prepare the following state

$$|\psi_0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle_L + e^{\frac{i\pi}{4}}|1\rangle_L\right).$$

This state can be prepared with a circuit of form given in Figure 2. For this purpose, let $\widetilde{U} = e^{\frac{i\pi}{4}}\sigma_x\sigma_z{\sigma_z}^{1/2}$ and $|\psi_1\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle_L - e^{\frac{i\pi}{4}}|1\rangle_L\right)$. Then $\widetilde{U}|\psi_0\rangle = |\psi_0\rangle$, $\widetilde{U}|\psi_1\rangle = -|\psi_1\rangle$, and $U_{\text{flip}} = \sigma_z$. Finally, see that $|0\rangle = \frac{1}{\sqrt{2}}(|\psi_0\rangle + |\psi_1\rangle)$. Note, as required in section 4.3, both $\widetilde{U}$ and U_{flip} are in the directly fault tolerant set. Hence we have all the requirements of the previous section, and thus we may use that method to create $|\psi_0\rangle$.

Now we are ready to describe the fault tolerant ${\sigma_z}^{1/4}$ without measurement. The circuit in Figure 3 shows the fault tolerant implementation of ${\sigma_z}^{1/4}$ on a codeword $|x\rangle_L$. In this circuit, $\mathcal{N}$ is the unitary operation defined in (1). Apart from replacing the standard measurements by the $\mathcal{N}$ circuit, this figure is exactly the same as the one drawn in [4] to implement the ${\sigma_z}^{1/4}$ gate. In this figure each input in fact denotes a block of qubits, and operations are bit-wise.

4.5. Fault tolerant Toffoli without measurement

The more conventional (and more complicated) set of universal fault tolerant gates contain the Toffoli in-

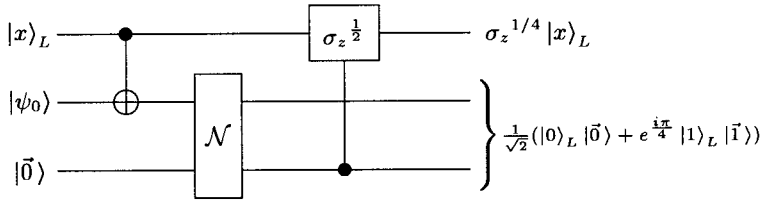

Figure 3. Fault tolerant $\sigma_z^{1/4}$ without measurement.

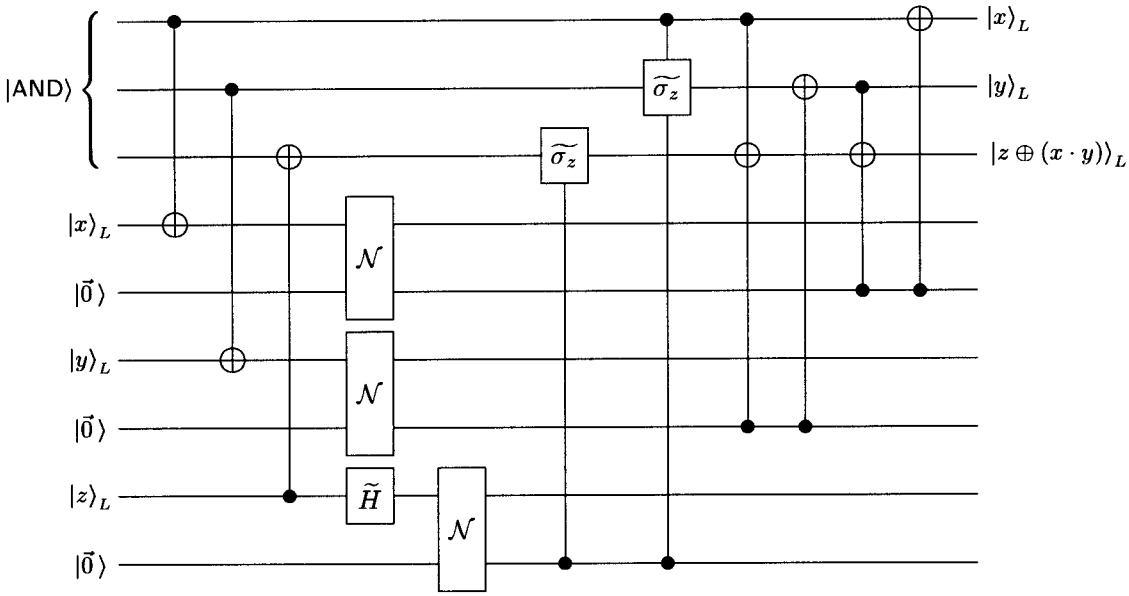

Figure 4. Fault tolerant Toffoli without measurement.

stead of the $\sigma_z^{1/4}$. We show explicitly how to implement Toffoli on encoded data without using any measurement. This scheme is a modified version of Shor's original method for implementing Toffoli on codewords [21], and is similar to the one applied to $\sigma_z^{1/4}$.

In Shor's method (as in the other basis we have shown) a preparation of a special state is required, hence we first prepare the state

$$|\mathsf{AND}\rangle = \tfrac{1}{2}\left(|000\rangle_L + |010\rangle_L + |100\rangle_L + |111\rangle_L\right), \qquad (2)$$

without using measurement, based on our scheme presented in Section 4.3..

To get $|\mathsf{AND}\rangle$ we let $U = \Lambda(\sigma_z) \otimes \sigma_z$, and we chose

$$\left|\overline{\mathsf{AND}}\right\rangle = \tfrac{1}{2}\left(|001\rangle_L + |011\rangle_L + |101\rangle_L + |110\rangle_L\right).$$

Then $\widetilde{U}\,|\mathsf{AND}\rangle = |\mathsf{AND}\rangle$, $\widetilde{U}\left|\overline{\mathsf{AND}}\right\rangle = -\left|\overline{\mathsf{AND}}\right\rangle$, $U_{\mathsf{flip}} = I \otimes I \otimes \sigma_x$, and

$$\tfrac{1}{\sqrt{2}}\left(|\mathsf{AND}\rangle + \left|\overline{\mathsf{AND}}\right\rangle\right) = (\widetilde{H} \otimes \widetilde{H} \otimes \widetilde{H})\,|000\rangle_L\,.$$

Note, as required in section 4.3, both $\widetilde{U}$ and U_{flip} are in the directly fault tolerant set. Hence we have all the requirements of the previous section, and thus we may use that method to create $|\mathsf{AND}\rangle$.

A different solution to this step was given (independently) by D. Aharonov and M. Ben-Or [2]. Our procedure for constructing the fault tolerant Toffoli gate is presented in Figure 4. In this circuit $\mathcal{N}$ is the unitary operation defined in (1); apart from replacing the standard measurements by our $\mathcal{N}$ circuit, this figure is exactly the same as the one drawn by Preskill [19] to describe Shor's way of obtaining the Toffoli gate. Note that in this figure each input represents a block of qubits and operations on these blocks are defined in the natural way. Also note that the first three top outputs of this circuit are in a tensor product with the rest of the outputs.

5. Error recovery in the error correction process

Standard error correction can be viewed as a computation with more than one good answer, and thus

belongs to case (1) discussed in Section 2. On different computers in the ensemble the syndrome of the error will be different, and thus is not unique. In the standard error correction prescription, measurement is used to collapse the ancilla qubits containing the information as to which error occurred (the syndrome). Then these syndrome bits are processed by a classical reversible algorithm to determine the errors, and a unitary operation to correct the error is applied to the data qubits by the output bits of the classical algorithm. In the measurement-free case, the ancilla qubits need not be measured, and the classical subroutine (following the measurement) could be incorporated into the original quantum algorithm.

The standard error correction operations require the use of a universal set of classical gates (e.g. NOT, CNOT, Toffoli). As in Section 3, for the classical part of the computation we do not care about phase errors, and as such we do not need the full power of quantum fault tolerance in this part of the computation. Hence, the techniques of Section 3 can be applied so that the classical subroutine is carried out on a classical code. The state of the ancilla qubits can be first copied onto a classical repetition code using the $\mathcal{N}$ gate. Now classical reversible computation can be performed on the repetition code and then a control operation can be performed on the quantum data to correct for the errors.

Since phase errors from the classical sub-circuit will not propagate to the quantum data, using repetition codes to correct for any bit errors in the sub-circuit is sufficient. The observation that phase errors cannot propagate from the "classical" part of the computation allows one to fault tolerantly replace quantum Toffoli gates by classical ones in the error recovery process.

6. Concluding Remarks

To summarize, we showed that running algorithms on bulk (ensemble) computers is not always straightforward. We modified fault tolerance protocols so that they can run on ensemble computers, such as NMR quantum computers, where individual qubit measurement is not available.

In a prior work, addressing fault tolerant computation, Aharonov and Ben–Or [3] have observed that the measurements required for fault tolerant computation can be substituted by reversible classical circuits performing controlled operations. D. Aharonov also sent us a manuscript[1] with results regarding Toffoli gate which are very similar to those obtained here. Knill, Laflamme, and Zurek [15] followed a different approach that potentially does not require measurements. However, to the best of our knowledge, a proof of universal fault tolerant computation via their approach is not available. In particular, a measurement-free implementation of the Hadamard gate using that approach has not been demonstrated. Finally, Peres [18] also discusses the possibility of measurement–free encoding and decoding procedures in quantum error–correction. However, in his scheme the quantum information is transformed to a single qubit, and his method is not suitable for fault tolerant computation.

We are thankful to Dorit Aharonov for many helpful remarks.

This work was supported in part by grants from the Revolutionary Computing group at JPL (contract #961360), and from the DARPA Ultra program (subcontract from Purdue University #530–1415–01). The work of T.M. was supported in part by the Israeli Ministry of Defense.

References

[1] D. Aharonov. Private communication.

[2] D. Aharonov and M. Ben-Or. Fault tolerant quantum computation with constant error rate. submitted to SIAM journal of computation. quant-ph/9906129.

[3] D. Aharonov and M. Ben-Or. Polynomial simulations of decohered quantum computers. In *Proceedings of 37th Annual Symposium on Foundations of Computer Science, FOCS'96*, pages 46–55. IEEE Computer Society Press, Los Alamitos, CA, USA, 1996.

[4] P. O. Boykin, T. Mor, M. Pulver, V. P. Roychowdhury, and F. Vatan. On universal and fault–tolerant quantum computation. In *Proc. of 40th Ann. Symp. on Found. of Comp. Sc.*, pages 486–494, 1999. quant-ph/9906054.

[5] P. O. Boykin, T. Mor, M. Pulver, V. P. Roychowdhury, and F. Vatan. A new universal and fault-tolerant quantum basis. *Information Processing Letters*, 75:101–107, 2000.

[6] P. O. Boykin, T. Mor, V. P. Roychowdhury, and F. Vatan. Algorithms on ensemble quantum computers. quant-ph/9907067, 1999.

[7] P. O. Boykin, T. Mor, V. P. Roychowdhury, F. Vatan, and R. Vrijen. Algorithmic cooling of ensemble computers. *Proceedings of the National Academy of Sciences*, 99(6):3388–3393, 2002.

[8] G. Brassard, S. L. Braunstein, and R. Cleve. Teleportation as a quantum computation. *Physica D*, 120(1–2):43–47, 1 September 1998.

[9] D. Cory, M. Price, W. Mass, E. Knill, R. Laflamme, W. Zurek, T. Havel, and S. Somaroo. Experimental quantum error correction. *Phys. Rev. Lett.*, 81:2152–2155, 1998.

[10] D. G. Cory, A. F. Fahmy, and T. F. Havel. Ensemble quantum computing by nuclear magnetic resonance spectroscopy. *Proceedings of the National Acadamy of Sciences of USA*, 94:1634–1639, 1997.

[11] N. Gershenfeld and I. L. Chuang. Bulk spin-resonance quantum computation. *Science*, 275(5298):350–356, 17 January 1997.

[12] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing, STOC'96*, pages 212–219. ACM Press, 1996.

[13] J. A. Jones, M. Mosca, and R. H. Hansen. Implementation of a quantum search algorithm on a quantum computer. *Nature*, 393:344–346, 1998.

[14] A. Yu. Kitaev. Quantum computation: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997.

[15] E. Knill, R. Laflamme, and W. H. Zurek. Accuracy threshold for quantum computation. quant-ph/9610011, 1996.

[16] E. Knill, R. Laflamme, and W. H. Zurek. Resilient quantum computation: error models and threshold. *Proceedings of the Royal Society of London, Series A*, 454(1969):365–384, 8 January 1998.

[17] M. A. Nielsen, E. Knill, and R. Laflamme. Complete quantum teleportation using nuclear magnetic resonance. *Nature*, 396:52–55, 1998.

[18] A. Peres. Quantum disentanglement and computation. *Superlattices and Microstructures*, 23(3–4):373–379, 1998.

[19] J. Preskill. Reliable quantum computers. *Proceedings of the Royal Society of London, Series A*, 454(1969):385–410, 8 January 1998.

[20] L. J. Schulman and U. Vazirani. Scalable nmr quantum computation. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing, STOC'99*, pages 322–329. IEEE Computer Society Press, 1999. quant-ph/9804060.

[21] P. W. Shor. Fault-tolerant quantum computation. In *Proceedings of 37th Annual Symposium on Foundations of Computer Science, FOCS'96*, pages 56–65. IEEE Computer Society Press, Los Alamitos, CA, USA, 1996.

[22] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.

[23] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang. Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414:883–887, 2001.